AR

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/746,796	12/22/2000	Ariel Cohen	00-162 1496.0047	9162

24319 7590 02/22/2008

LSI CORPORATION
1621 BARBER LANE
MS: D-106
MILPITAS, CA 95035

EXAMINER	
DOLLINGER, TONIA LYNN MEONSKE	
ART UNIT	**PAPER NUMBER**
2181	

MAIL DATE	DELIVERY MODE
02/22/2008	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

Commissioner for Patents
United States Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450
www.uspto.gov

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Application Number: 09/746,796
Filing Date: December 22, 2000
Appellant(s): COHEN ET AL.

Christopher Maiorana, Registration No. 42,829
For Appellant

EXAMINER'S ANSWER

This is in response to the appeal brief filed May 4, 2007 appealing from the Office action mailed May 18, 2004

(1) Real Party in Interest

A statement identifying by name the real party in interest is contained in the brief.

(2) Related Appeals and Interferences

The examiner is not aware of any related appeals, interferences, or judicial proceedings which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

(3) Status of Claims

The statement of the status of claims contained in the brief is correct.

(4) Status of Amendments After Final

The appellant's statement of the status of amendments after final rejection contained in the brief is correct.

No amendment after final has been filed.

The summary of claimed subject matter contained in the brief is correct.

(5) Summary of Claimed Subject Matter

The summary of claimed subject matter contained in the brief is correct.

(6) Grounds of Rejection to be Reviewed on Appeal

The appellant's statement of the grounds of rejection to be reviewed on appeal is substantially correct. The changes are as follows: Claim 22 is rejected under under 35 U.S.C. 102(b) as being anticipated by Hilgendorf et al, U.S. Patent Number 5,925,124. The rejection of claim 22 appears on page 6 of the Final Rejection mailed on May 18, 2004. A typo on page 2 of the Final Rejection inadvertently omitted claim 22 from the heading.

(7) Claims Appendix

The copy of the appealed claims contained in the Appendix to the brief is correct.

(8) Evidence Relied Upon

5,925,124	Hilgendorf et al.	7-1999
4,439,828	Martin	3-1984
6,374,286	Gee et al.	4-2002

(9) Grounds of Rejection

The following ground(s) of rejection are applicable to the appealed claims:

Claim Rejections - 35 USC § 102

1. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States

2. Claims 1-6, 8-9, 12-14, 17-19, and 22 are rejected under 35 U.S.C. 102(b) as being anticipated by Hilgendorf et al, U.S. Patent Number 5,925,124 (herein referred to as Hilgendorf).

3. Referring to claim 1 Hilgendorf has taught an apparatus comprising:

a. a circuit *(the entire circuit shown in Figure 1)*

b. configured to translate *(column 3, lines 38-44, see "instruction translation")*

c. instruction codes of a first instruction set *(Figure 1, column 1, lines 37-41, column 4, lines 32-47, The CISC instruction set of Hilgendorf et al. is the claimed first instruction set. An instruction code is an operation code, or op-code (see the Microsoft Computing Dictionary, fourth edition, page 238, for extrinsic evidence of this fact.). Therefore the op-codes of the CISC instruction set in Hilgerdorf, element 102, are the claimed instruction codes of a first instruction set.)*

d. on-the-fly *(column 3, lines 38-44, On-the-fly is interpreted to be at run-time. The translation into addresses occurs at run-time, which is on-the-fly.)*

e. into addresses into a microcode memory *(Hilgendorf, abstract, Figures 1 and 2, reference number 104 and 106 of figure 1, column 3 lines 8-56, column 7 lines 9-30; The op-code of the first instruction set is translated to addresses used to indicate a set of internal instructions in the translation table, which is implemented using memory i.e. ROM or RAM, thus the translation table is a microcode memory.)*

f. containing sequences of instruction codes of a second instruction set *(column 11, lines 61-64, Figure 1, An instruction code is an operation code, or op-code, for support see the Microsoft Computer Dictionary, fourth edition, page 238. Element 105 contains an internal sequence of instruction codes, or op-codes, 1/2/3, element 107 for code B, or*

the claimed second instruction set. The other several hundred entries in the translation table element 106, or microcode memory, also contain sequences of internal instruction codes of the second instruction set, or RISC instruction set, see column 8, lines 60-column 9, line 3.)

g. that emulate a functionality of the instruction codes of the first instruction set *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, The CISC instructions codes are translated to RISC instruction codes. The translated RISC instruction codes emulate the functionality of the CISC instruction codes. Column 5, lines 58-67. Several internal RISC instructions emulate a CISC external instruction.).*

4. Referring to claim 2, Hilgendorf has taught the apparatus of claim 1, as described above, and wherein said sequences of instruction codes of said second instruction set generated in response to said instruction codes of said first instruction set are stored in a cache *(Hilgendorf, abstract, Figures 1 and 2,element 106, column 3 lines 8-56, column 7 lines 9-17).*

5. Referring to claim 3, Hilgendorf has taught the apparatus of claim 1, as described above, and wherein said circuit comprises a decoder configured to generate said addresses into said microcode memory *(Hilgendorf, abstract, Figures 1 and 2, reference numbers 104 and 106 of Figure 1, column 3 lines 8-56, column 7 lines 9-30).*

6. Referring to claim 4, Hilgendorf has taught the apparatus of claim 1, as described above, and wherein predetermined sequences of said instruction codes of said first instruction set are used to address said microcode memory *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, column 7 lines 9-30, The translation table is designed to have any and all predetermined sequences, or programs, of the First Instruction set address the table for instruction translation.*

A program for a computer system that for all of eternity only performs one single instruction is not useful. A program for a computer system must have a sequence of several instructions. Any sequence of instructions contains a plurality of subsequences. So a program necessarily contains several sequences. Therefore Hilgendorf has in fact taught predetermined sequences of said instruction codes of said first instruction set (a plurality of subsequences in a program) are used to address said microcode memory.).

7. Referring to claim 5, Hilgendorf has taught the apparatus of claim 1, as described above, and wherein addresses into said microcode memory are generated by a look-up-table in response to said instruction codes of said first instruction set *(Hilgendorf, abstract Figures 1 and 2, column 3 lines 8-56, column 7 lines 9-30, elements 103 and 104).*

8. Referring to claim 6, Hilgendorf has taught the apparatus of claim 1, as described above, and wherein said instruction codes of said second instruction set comprise native instructions of a target processor *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, column 7 lines 9-17, column 12 lines 17-22; The instructions that are to be executed must be of the same architecture as the host computer, i.e. the instructions must be native to the host, or target, computer. If the instructions are not native to the host processor then the instructions will not be able to execute on the host because the instructions are designed for some other computer architecture.).*

9. Referring to claim 8, Hilgendorf has taught the apparatus of claim 3, as described above, and wherein said microcode memory can be reprogrammed to support different processors *(Hilgendorf, column 2, lines 53-64, column 3 lines 45-56, column 4 lines 51-61).*

10. Referring to claim 9, Hilgendorf has taught the apparatus of claim 1, as described above, and wherein said circuit is configured to format the sequences of instruction codes of said second instruction set according to an opcode format of a processor *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, column 9 lines 42-59, column 12 lines 17-38, The instructions that are to be executed must be of the same architecture, or native, to the host computer which includes the operation code, or opcode, format.)*.

11. Referring to claim 12, Hilgendorf has taught the apparatus of claim 1, as described above, and wherein said circuit comprises a native instruction sequence generator circuit *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, column 9 lines 42-59, column 11, lines 53-67, column 12 lines 17-38, The generated translated instructions that are to be executed must be of the same architecture as the host computer.)*.

12. Referring to claim 13, Hilgendorf has taught the apparatus of claim 1, as described above, and wherein said circuit is coupled between processor and a memory system *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, column 12 lines 17-38, column 6 lines 54-61, The external instructions come from memory and the internal instructions are sent on to the execution unit to be processed.)*.

13. Referring to claim 14, Hilgendorf has taught the apparatus of claim 13, as described above, and wherein said circuit is configured to (i) directly connect said processor and said memory system during a first state of operation and (ii) during a second state of operation, communicate with said processor as though said circuit was the memory system and communicate with said memory system as though said circuit was the processor *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, column 12 lines 17-38, column 6 lines 54-61, The*

external instructions come from memory and the internal instructions are sent on to the execution unit to be processed).

14. Referring to claim 17, Hilgendorf has taught an apparatus comprising:

a. means *(the entire circuit shown in Figure 1)*

b. for translating *(column 3, lines 38-44, see "instruction translation")*

c. instruction codes of a first instruction set (*Figure 1, column 1, lines 37-41, column 4, lines 32-47, The CISC instruction set of Hilgendorf et al. is the claimed first instruction set. An instruction code is an operation code, or op-code (see the Microsoft Computing Dictionary, fourth edition, page 238, for extrinsic evidence of this fact.). Therefore the op-codes of the CISC instruction set in Hilgerdorf, element 102, are the claimed instruction codes of a first instruction set.)*

d. on-the-fly *(column 3, lines 38-44, On-the-fly is interpreted to be at run-time. The translation into addresses occurs at run-time, which is on-the-fly.)*

e. into addresses into a microcode memory *(Hilgendorf, abstract figures 1 and 2, reference number 104 and 106 of figure 1, column 3 lines 8-56, column 7 lines 9-30; The op-code of the first instruction set is translated to addresses used to indicate a set of internal instructions in the translation table, which is implemented using memory i.e. ROM or RAM, thus the translation table is a microcode memory.)*

f. containing sequences of instruction codes of a second instruction set *(column 11, lines 61-64, Figure 1. An instruction code is an operation code, or op-code, for support see the Microsoft Computer Dictionary, fourth edition, page 238. , Element 105 contains an internal sequence of instruction codes, or op-codes. 1/2/3 for code B, or the claimed*

second instruction set. The other several hundred entries in the translation table element 106, or microcode memory, also contain sequences of internal instruction codes of the second instruction set, or RISC instruction set, see column 8, lines 60-column 9, line 3.)

g. that emulate a functionality of the instruction codes of said first instruction set *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, The CISC instructions codes are translated to RISC instruction codes. The translated RISC instruction codes emulate the functionality of the CISC instruction codes. Column 5, lines 58-67. Several internal RISC instructions emulate a CISC external instruction.);*

h. means for receiving said instruction codes of said first instruction set *(Hilgendorf, Figures 1 and 2, abstract, column 1, lines 37-41, column 4, lines 32-47, column 3 lines 8-56, column 12 lines 17-38, column 6 lines 54-61, The CISC instruction set of Hilgendorf et al. is the claimed first instruction set. An instruction code is an operation code, or op-code (see the Microsoft Computing Dictionary, fourth edition, page 238, for extrinsic evidence of this fact.). Element 104 receives the external instructions of the first instruction set from memory that contain the instruction codes.);* and

i. means for presenting said sequence of instruction codes of said second instruction set *(Hilgendorf, column 9 lines 14-19).*

15. Referring to claim 18 Hilgendorf has taught a method for

a. on-the-fly *(column 3, lines 38-44, On-the-fly is interpreted to be at run-time. The translation into addresses occurs at run-time, which is on-the-fly.)*

b. translation of instructions *(column 3, lines 38-44, see "instruction translation")*

c. of a first instruction set *(Figure 1, column 1, lines 37-41, column 4, lines 32-47, The CISC instruction set of Hilgendorf is the claimed first instruction set.)*

d. into instructions of a second instruction set *(Figure 1, column 1, lines 37-41, column 4, lines 32-47, The RISC instruction set of Hilgendorf is the claimed second instruction set.)* comprising the steps of:

i. receiving an instruction code of said first instruction set *(Hilgendorf, Figures 1 and 2, abstract, column 1, lines 37-41, column 4, lines 32-47, column 3 lines 8-56, column 12 lines 17-38, column 6 lines 54-61, The CISC instruction set of Hilgendorf et al. is the claimed first instruction set. An instruction code is an operation code, or op-code (see the Microsoft Computing Dictionary, fourth edition, page 238, for extrinsic evidence of this fact.). Element 104 receives the external instructions of the first instruction set from memory that contain the instruction codes.):*

ii. generating an address *(Figure 1, element 103 and 104) into a microcode memory (Hilgendorf, abstract, Figures 1 and 2, reference number 104 and 106 of figure 1, column 3 lines 8-56, column 7 lines 9-30; The op-code of the first instruction set is translated to addresses used to indicate a set of internal instruction opcodes, or microcode, in the translation table. The table is implemented using memory i.e. ROM or RAM, thus the translation table is a microcode memory.)*

iii. in response to said instruction code of said first instruction set using a hardware translator *(column 3, lines 38-44, Figures 1 and 2, see "instruction translation")*,

iv. wherein said address points to a sequence of: instruction codes of said second instruction set *(column 11, lines 61-64, Figure 1, An instruction code is an operation code, or op-code, for support see the Microsoft Computer Dictionary, fourth edition, page 238. The address, element 103, points to element 105, which contains an internal sequence of instruction codes, or op-codes, 1/2/3 for code B, or the claimed second instruction set.)*

v. that will emulate said instruction code of said first instruction set *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, The CISC instructions codes are translated to RISC instruction codes. The translated RISC instruction codes emulate the functionality of the CISC instruction codes. Column 5, lines 58-67. Several internal RISC instructions emulate a CISC external instruction.)*; and

vi. presenting said sequence of instruction codes of said second instruction set *(Hilgendorf, column 9 lines 14-19)*.

16. Referring to claim 19 Hilgendorf has taught wherein step B comprises the sub-step of:

a selecting said address from a look-up table in response to said instruction code of said first instruction set *(Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, column 7 lines 9-30)*.

17 Referring to claim 22 Hilgendorf has taught wherein said microcode memory further comprises one or more of

a. (i) a size for each sequence of instruction codes of said second instruction set *(Hilgendorf, Figure 3, element 300, column 8 line 65-column 9 line 3),*

b. (ii) a value representing how many bytes an instruction uses from said instruction codes of said first instruction set *(Since the claim states "one or more" this limitation does not need to be present in the prior art for the art to read on the claimed invention.),*

c. and (iii) a stack change variable indicating whether the stack increases of decreases due to said instruction codes of said first instruction set and by how much *(Since the claim states "one or more" this limitation also does not need to be present in the prior art for the art to read on the claimed invention.).*

Claim Rejections - 35 USC § 103

18. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

19. Claims 10, 11, and 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hilgendorf in view of Martin U.S. Patent Number 4,439,828 (herein referred to as Martin).

20. Referring to claims 10, 11, and 20 Hilgendorf has not taught wherein said circuit is configured to detect optimizable sequences of instruction codes on-the-fly. However, Hilgendorf recognizes the need to optimize instruction execution by exploiting instruction parallelism with

superscalar processors to increase performance *(Hilgendorf, column 1, line 56-column 2, line 63)*. Martin has taught wherein said circuit is configured to detect optimizable sequences of instruction codes on-the-fly *(Martin abstract column 2 lines 20-52)*. It would have been obvious to one of ordinary skill in the art at the time the invention was made to have the circuit of Hilgendorf be configured to detect optimizable sequences of instruction codes on-the-fly. Since Martin shows us that one instruction takes the place of two or more instructions, and that this function improves performance *(Martin column 2 lines 20-30)*, by allowing other locations in the instruction buffer to be open, and by allowing for a single instruction to take the place of multiple instructions, the processor would also save time. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have the processor of Hilgendorf achieve its goal of improving system performance by configuring the circuit to detect optimizable sequences of instruction codes on-the-fly, as taught by Martin *(Martin column 2 lines 20-30)*.

21. Claims 7 is rejected under 35 U.S.C. 103(a) as being unpatentable over Hilgendorf. Hilgendorf has not explicitly taught wherein said target processor is selected from the group consisting of MIPS, ARM, and Motorola 68K. However, Hilgendorf has taught that the translation circuit translates the first instruction code into many different codes by replacing the translation table's contents *(Hilgendorf column 4 lines 51-61)* Since the MIPS, ARM, and Motorola 68K are all commonly used processors in the art, one of ordinary skill in the art at the time of the invention would have recognized that one would use this invention to translate a plurality of instructions from one code type to one of the code types of the MIPS, ARM, and Motorola 68K processors. Therefore, it would have been obvious to one of ordinary skill in the

art at the time of the invention would have made the translation table translate instruction to a format to one of the MIPS, ARM, or Motorola 68K processors since these processors are commonly used in industry and would be widely available for use in many systems.

22. Claims 15 is rejected under 35 U.S.C. 103(a) as being unpatentable over Hilgendorf with US Patent 6,374,286, Gee et al , US Patent 6,615,340 (hereinafter referred to as Gee), Wilmot, II, US Patent 5,926,642, Favor, and US Patent 6,502,109, Aravamudan et al. cited for extrinsic evidence. Hilgendorf has taught the apparatus of claim 1. Hilgendorf has not explicitly taught wherein said instruction codes of said first instruction set comprise Java bytecodes. However, Hilgendorf has taught that the translation circuit translates the first instruction code into many different codes by replacing the translation table's contents *(Hilgendorf column 3 lines 45-56, column 4 lines 51-61)*. One such possible instruction code is Java. The Java language has been quickly embraced by the computer software community and has been demonstrated to be efficient and robust for a wide variety of general purpose computing applications *(Gee, Column 2, lines 28-37)*. For example, Hilgendorf's invention is part of a superscalar processor Superscalar processors are typically employed as webs servers because of the high performance associated with the superscalar design *(For extrinsic evidence of this fact, please see US Patent 6,615,340, Wilmot, II, column 1, lines 10-23, and US Patent 5,926,642, Favor, Columns 60-62, specifically see column 62, lines 41-45.)*. A common way of executing programs on the world wide web is by using java bytecodes because using java bytecodes enables the execution of programs across any platform so long as an interpreter, or translator, is present *(For extrinsic evidence of this fact, please see US Patent 6,502,109, Aravamudan et al., column 2, lines 45-65)*.

23. Since superscalar computers are typically employed as web servers, one of ordinary skill in the art at the time the invention was made would have recognized to employ Hilgendorf's superscalar computer as a web server because of the high performance associated with the superscalar design. Furthermore since Java is a commonly used programming language for the execution of programs on the world wide web, one of ordinary skill in the art at the time the invention was made would have recognized the desirability of executing Java bytecodes on a web server that includes Hilgendorf's invention since most of the programs available on the world wide web are already written in Java bytecodes. In order to execute these widely available Java bytecodes in Hilgendorf's superscalar processor, the Java bytecodes must be translated to the host's instruction code. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have the superscalar web server of Hilgendorf translate Java bytecodes into the host's computer code, since Java is a popular and widely used programming language, and many programs available on the world wide web would already be written in Java bytecodes.

24. Claims 16 and 21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hilgendorf in view of Gee et al., U.S. Patent Number 6,374,286 (herein referred to as Gee) with US Patent 6,615,340, Wilmot, II, US Patent 5,926,642, Favor, and US Patent 6,502,109, Aravamudan et al. cited for extrinsic evidence.

25. Referring to claim 16, Hilgendorf has taught the apparatus according to claim 1. Hilgendorf has not taught wherein said circuit comprises a portion of a Java virtual machine, i.e. where instruction codes of said first instruction set comprise Java bytecodes. However, Hilgendorf has taught that the translation circuit translates the first instruction code into many

different codes by replacing the translation table's contents *(Hilgendorf column 3 lines 45-56, column 4 lines 51-61)*. One such possible instruction code is Java. The Java language has been quickly embraced by the computer software community and has been demonstrated to be efficient and robust for a wide variety of general purpose computing applications *(Gee, Column 2, lines 28-37)*. For example, Hilgendorf's invention is part of a superscalar processor. Superscalar processors are typically employed as webs servers because of the high performance associated with the superscalar design *(For extrinsic evidence of this fact, please see US Patent 6,615,340, Wilmot, II, column 1, lines 10-23, and US Patent 5,926,642, Favor, Columns 60-62, specifically see column 62, lines 41-45.)*. A common way of executing programs on the world wide web is by using java bytecodes because using java bytecodes enables the execution of programs across any platform so long as an interpreter, or translator, is present *(For extrinsic evidence of this fact, please see US Patent 6,502,109, column 2, lines 45-65)*. This translation is performed by a Java Virtual machine, which allows for a flexible run-time environment *(Gee, column 2, lines 38-47)*.

26. Since superscalar computers are typically employed as web servers, one of ordinary skill in the art at the time the invention was made would have recognized to employ Hilgendorf's superscalar computer as a web server because of the high performance associated with the superscalar design. Furthermore since Java is a commonly used programming language for the execution of programs on the world wide web, one of ordinary skill in the art at the time the invention was made would have recognized the desirability of executing Java bytecodes on a web server that includes Hilgendorf's invention since most of the programs available on the world wide web are already written in Java bytecodes. In order to execute these widely available

Java bytecodes in Hilgendorf's superscalar processor, the Java bytecodes must be translated to the host's instruction code. One of ordinary skill in the art would have also recognized to perform this translation with a Java Virtual machine to allow for a flexible run-time environment. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have the superscalar web server of Hilgendorf translate Java bytecodes into the host's computer code using a flexible run-time Java Virtual Machine since Java is a popular and widely used programming language, and many programs available on the world wide web would already be written in Java bytecodes.

27. Hilgendorf has also not taught a Java Virtual Machine implemented in hardware. However, Gee has taught that implementing the Java Virtual Machine in hardware *(i.e. direct execution JAVA processor)* reduces the drawbacks of a JVM not implemented in hardware, such as speed and difficulty in meeting strict timing requirements *(Gee, column 2, lines 38-47)*. Thus allowing for a higher instruction throughput, and the ability to complete instructions more quickly, which is important with real-time applications. Therefore it would have been obvious to one of ordinary skill in that art at the time the invention was made to have the JVM implemented in hardware, as taught by Gee, for the desirable purpose of executing the Java bytecodes faster.

28. Referring to claim 21 Hilgendorf has not taught wherein said sequences of instruction codes of said instruction set comprise one or more virtual stack references. However, Hilgendorf has taught that the translation circuit translates the first instruction code into many different codes by replacing the translation table's contents *(Hilgendorf column 3 lines 45-56, column 4 lines 51-61)*. One such possible instruction code is Java. The Java language has been quickly embraced by the computer software community and has been demonstrated to be efficient and

robust for a wide variety of general purpose computing applications *(Gee, Column 2, lines 28-37)*. For example, Hilgendorf's invention is part of a superscalar processor. Superscalar processors are typically employed as webs servers because of the high performance associated with the superscalar design *(For extrinsic evidence of this fact, please see US Patent 6,615,340, Wilmot, II, column 1, lines 10-23, and US Patent 5,926,642, Favor, Columns 60-62, specifically see column 62, lines 41-45.)*. A common way of executing programs on the world wide web is by using java bytecodes because using java bytecodes enables the execution of programs across any platform so long as an interpreter, or translator, is present *(For extrinsic evidence of this fact, please see US Patent 6,502,109, column 2, lines 45-65)*. This translation is performed by a Java Virtual machine, which allows for a flexible run-time environment *(Gee, column 2, lines 38-47)*

29. Since superscalar computers are typically employed as web servers, one of ordinary skill in the art at the time the invention was made would have recognized to employ Hilgendorf's superscalar computer as a web server because of the high performance associated with the superscalar design. Furthermore since Java is a commonly used programming language for the execution of programs on the world wide web, one of ordinary skill in the art at the time the invention was made would have recognized the desirability of executing Java bytecodes on a web server that includes Hilgendorf's invention, since most of the programs available on the world wide web are already written in Java bytecodes. In order to execute these widely available Java bytecodes in Hilgendorf's superscalar processor, the Java bytecodes must be translated to the host's instruction code. One of ordinary skill in the art would have also recognized to perform this translation with a Java Virtual machine to allow for a flexible run-time environment. Therefore, it would have been obvious to one of ordinary skill in the art at the time

the invention was made to have the superscalar web server of Hilgendorf translate Java bytecodes into the host's computer code using a flexible, run-time, Java Virtual Machine since Java is a popular and widely used programming language, and many programs available on the world wide web would already be written in Java bytecodes.

30. When the Hilgendorf reference translates Java code with a Java Virtual Machine, as taught by Gee, the sequences of instruction codes of said instruction set necessarily comprise one or more virtual stack references *(Gee, Column 7, lines 1-10)* because in order to implement Java with a Java Virtual Machine it is necessary to have virtual stack references

(10) Response to Argument

Issue	Appeal Brief	Appellant's Position	Examiner's Response
Rejection of Claims 1-3, 5-9, 12-14 and 17-19	Pages 7-10	Hilgendorf does not disclose or suggest a microcode memory containing sequences of instruction codes of a second instruction set that emulate a functionality of the instruction codes of the first instruction set, as presently claimed. In particular, Hilgendorf refers to **converting** microcode instructions of a code A into internal instructions of a code B, **not translating** instruction codes of a first instruction set on-the-fly **into addresses into a microcode memory containing sequences of instruction codes of a second instruction set** that emulate a functionality of the instruction codes of the first instruction set, as presently claimed.	Hilgendorf has taught the translation of external CISC instructions into internal RISC instructions. (See column 4 lines 32-34). Applicant is correct in that Hilgendorf has taught **converting** microcode instructions of a code A into internal instructions of a code B. However, this conversion is necessarily a translation. Hilgendorf in fact uses the term conversion and translation interchangeable in the specification. Specifically see column 3, lines 42-44 and column 6, lines 54-57. Furthermore, the Microsoft Computer Dictionary, fourth edition, defines "translate" as "In programming, to **convert** a program from one language to another. ..." Furthermore, Hilgendorf has in fact taught instruction codes of a first instruction set (Figure 1, column 1, lines 37-41, column 4, lines 32-47, The CISC instruction set of Hilgendorf et al. is the claimed first instruction set. An instruction code is an operation code, or op-code (see the Microsoft Computing Dictionary, fourth edition, page 238, for extrinsic evidence of this fact.). Therefore the op-codes of the CISC instruction set in Hilgerdorf, element 102, are the claimed instruction codes of a first instruction set.) Therefore, Hilgendorf has in fact taught **translating**

		instruction codes of a first instruction set. This conversion, or translation, of the first instruction set is performed during run-time, which is on-the-fly (column 3, lines 38-44). The first instruction set (Figure 1, element 100, specifically element 102) is translated into addresses (Figure 1, element 103) by the address generation logic (Figure 1, element 104, column 7, lines 21-25). These address are sent into a microcode memory (Figure 1, element 106, column 7, lines 24-30, The address is used to access the translation table, or microcode memory. The translation table is a microcode memory because the table is a memory i.e. RAM or ROM, that stores microcodes, or instruction opcodes.) The microcode memory contains sequences of instruction codes of a second instruction set (column 7, lines 24-30, column 11, lines 61-64, Figure 1, An instruction code is an operation code, or op-code, for support see the Microsoft Computer Dictionary, fourth edition, page 238. The translation table, element 105, contains sequences of instruction opcodes for the internal RISC instruction set, 1/2/3 for code B, or the claimed second instruction set. The other several hundred entries in the translation table element 106, or microcode memory, also contain sequences of internal instruction codes of the second

			instruction set, or RISC instruction set, see column 8, lines 60-column 9, line 3. These opcodes are forwarded to generate the internal RISC instructions.) The instruction codes of the internal, or RISC, instruction set emulates a functionality of the instruction codes of the first instruction set, or CISC (Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, The CISC instructions codes are translated to RISC instruction codes. The translated RISC instruction codes emulate the functionality of the CISC instruction codes. Column 5, lines 58-67. Several internal RISC instructions emulate a CISC external instruction.) Therefore, Hilgendorf has taught a microcode memory (Figure 1, element 106) containing sequences of instruction codes of a second instruction set (column 7, lines 24-30, RISC opcodes, element 107, 1/2/3) that emulate a functionality of the instruction codes of the first instruction set (Column 5, lines 58-67, Figure 1, element 100 and 102), as presently claimed. Therefore this argument is moot.
Rejection of Claims 1-3, 5-9, 12-14 and 17-19	Pages 7-8	(1) Specifically, the cited reference clearly explains that the translation table 106 does not contain sequences of instruction codes of a second instruction set that emulate a functionality of the instruction codes of the first instruction set, as presently claimed. Rather, Hilgendorf clearly states that the translation table contains rearrangement information.	(1) However, the translation table in Hilgendorf does contain sequences of instruction codes (column 11, lines 61-64, Figure 1, column 7, lines 24-30, The translation table contains sequences of instruction opcodes for the internal RISC instruction set. An instruction code is an operation code, or op-code, for support see the Microsoft Computer Dictionary, fourth edition, page 238. Element 105 contains an internal

		sequence of instruction codes, or op-codes, 1/2/3, element 107 for code B RISC, or the claimed second instruction set. The other several hundred entries in the translation table element 106, or microcode memory, also contain sequences of internal instruction codes of the second instruction set, or RISC instruction set, see column 8, lines 60-column 9, line 3.) of a second instruction set (RISC) that emulate a functionality of the instruction codes of the first instruction set (Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, The CISC instructions codes are translated to RISC instruction codes. The translated RISC instruction codes emulate the functionality of the CISC instruction codes. Column 5, lines 58-67. Several internal RISC instructions emulate a CISC external instruction.). Therefore this argument is moot.
	(2) Furthermore, Hilgendorf is silent about a microcode memory containing sequences of instruction codes of a second instruction set that emulate a functionality of the instruction codes of the first instruction set, as presently claimed.	(2) Furthermore, Hilgendorf is not silent about a microcode memory containing sequences of instruction codes of a second instruction set that emulate a functionality of the instruction codes of the first instruction set, as presently claimed. The translation table, element 106 in Figure 1, is the microcode memory (The translation table is a microcode memory because the table is a memory i.e. RAM or ROM, that stores microcodes, or instruction opcodes.). The translation table contains sequences of instruction codes, or opcodes, of a second instruction set, or RISC, that emulate a functionality of the instruction codes of the first instruction set (Column 5,

			lines 38-67, specifically line 67, The translated RISC instructions emulate the functionality of the CISC instructions. Several internal RISC instructions are required to emulate a CISC external instruction. Also see argument (1) above.), as presently claimed. Therefore this argument is moot.
Rejection of Claims 1-3, 5-9, 12-14 and 17-19	Page 9	Hilgendorf does not disclose or suggest a circuit configured to translate instruction codes of a first instruction set on-the-fly into addresses into a microcode memory containing sequences of instruction codes of a second instruction set that emulate a functionality of the instruction codes of the first instruction set, as presently claimed.	Hilgendorf has taught a circuit (the entire circuit shown in Figure 1) configured to translate (column 3, lines 38-44, see "instruction translation") instruction codes of a first instruction set (column 4, lines 32-47, CISC is the first instruction set. The opcodes, element 102, are the instruction codes of the first instruction set. For support see the Microsoft Computer Dictionary, fourth edition, page 238.) on-the-fly (column 3, lines 38-44, On-the-fly is interpreted to be at run-time. The translation into addresses occurs at run-time, which is on-the-fly.) into addresses (Figure 1, element 103) into a microcode memory (Figure 1, element 106, column 7, lines 24-30, The address is used to access the translation table, or microcode memory. The translation table is a microcode memory because the table is a memory i.e. RAM or ROM, that stores microcodes, or instruction opcodes.) containing sequences of instruction codes of a second instruction set (column 7, lines 24-30, Figure 1, The translation table contains sequences of instruction opcodes for the internal RISC instruction set, element 107, 1/2/3. These opcodes are forwarded to generate the internal RISC instructions.) that emulate a functionality of the instruction codes of the first instruction set (Hilgendorf, abstract, Figures 1 and 2, column 3 lines

			lines 58-67, specifically line 67, The translated RISC instructions emulate the functionality of the CISC instructions. Several internal RISC instructions are required to emulate a CISC external instruction. Also see argument (1) above.), as presently claimed. Therefore this argument is moot.
Rejection of Claims 1-3, 5-9, 12-14 and 17-19	Page 9	Hilgendorf does not disclose or suggest a circuit configured to translate instruction codes of a first instruction set on-the-fly into addresses into a microcode memory containing sequences of instruction codes of a second instruction set that emulate a functionality of the instruction codes of the first instruction set, as presently claimed.	Hilgendorf has taught a circuit (the entire circuit shown in Figure 1) configured to translate (column 3, lines 38-44, see "instruction translation") instruction codes of a first instruction set (column 4, lines 32-47, CISC is the first instruction set. The opcodes, element 102, are the instruction codes of the first instruction set. For support see the Microsoft Computer Dictionary, fourth edition, page 238.) on-the-fly (column 3, lines 38-44, On-the-fly is interpreted to be at run-time. The translation into addresses occurs at run-time, which is on-the-fly.) into addresses (Figure 1, element 103) into a microcode memory (Figure 1, element 106, column 7, lines 24-30, The address is used to access the translation table, or microcode memory. The translation table is a microcode memory because the table is a memory i.e. RAM or ROM, that stores microcodes, or instruction opcodes.) containing sequences of instruction codes of a second instruction set (column 7, lines 24-30, Figure 1, The translation table contains sequences of instruction opcodes for the internal RISC instruction set, element 107, 1/2/3. These opcodes are forwarded to generate the internal RISC instructions.) that emulate a functionality of the instruction codes of the first instruction set (Hilgendorf, abstract, Figures 1 and 2, column 3 lines

		the first instruction set are used to address the microcode memory as presently claimed.	or programs, of the First Instruction set address the table for instruction translation. A program for a computer system that for all of eternity only performs one single instruction is not useful. A program for a computer system must have a sequence of instructions contains a plurality of subsequences. So a program necessarily contains several sequences. Therefore Hilgendorf has in fact taught predetermined sequences of said instruction codes of said first instruction set (a plurality of subsequences in a program) are used to address said microcode memory as presently claimed. Therefore this argument is moot.
Rejection of claim 4	Page 12	The Abstract of Hilgendorf is silent regarding a sequence of the instruction codes of the first instruction set being used to address the microcode memory, as presently claimed. Rather, each reference to an instruction of code A is singular, thus indicating only one instruction of code A is involved at a time.	However, the fact that only one instruction of code A in Hilgendorf may be involved at one time is irrelevant as the claim is not limited as such. Any sort of timing requirements that Appellant is arguing with respect to the sequence of instruction codes addressing the microcode memory does not appear anywhere in the claims. Hilgendorf has in fact taught sequences of the instruction codes of the first instruction set being used to address the microcode memory as presently claimed. In Hilgendorf, abstract, Figures 1 and 2, column 3 lines 8-56, column 7 lines 9-30, the translation table is designed to have any and all sequences, or programs, of the First Instruction set address the table for instruction translation. A program for a computer system that for all of eternity only performs one single instruction is not useful. A program for a computer

			system must have a sequence of several instructions. Any sequence of instructions contains a plurality of subsequences. So a program necessarily contains several sequences. Therefore Hilgendorf has in fact taught sequences of said instruction codes of said first instruction set (a plurality of subsequences in a program) are used to address said microcode memory as presently clamed. Therefore this argument is moot.
Rejection of Claims 10, 11, and 20	Pages 17-21	The Examiner admits that with respect to claims 10, 11 and 20 Hilgendorf has not taught wherein the circuit is configured to detect optimizable sequences of instruction codes on-the-fly. Martin does not appear to cure the deficiencies of Hilgendorf. In particular, Martin appears to be silent regarding a circuit configured to detect optimizable sequences of instruction codes on-the-fly. ... Nowhere in the above text does Martin expressly mention detecting optimizable sequences of instruction codes on-the-fly, as presently claimed.	However, Martin has taught a circuit configured to detect optimizable sequences of instruction codes on-the-fly. Martin has taught that when pre-defined instruction sequences are detected, a substitute instruction is generated to replace the first instruction sequence for the desirable purpose of improving system performance (column 2, lines 19-30). Further, this detection is performed during execution time, or on-the-fly (column 2, lines 19-52). Therefore this argument is moot
Rejection of Claims 10, 11, and 20	Pages 17-21	The Examiner has provided a personal conclusion that "It would have been obvious to one of ordinary skill in the art at the time of the invention to be configured to detect optimizable sequences of instruction codes on-the-fly. Since Martin shows us that one instruction can take the place of two or more instructions, and that this function improves performance". Such a conclusory statement is not adequate to meet the burden to identify specifically	However, it is respectfully noted that the motivation to combine Hilgendorf and Martin is not merely a conclusory statement as Appellant suggests. The motivation to combine Hilgendorf and Martin appear directly in the references. While Hilgendorf may have not taught wherein said circuit is configured to detect optimizable sequences of instruction codes on-the-fly, Hilgendorf recognizes the need to optimize instruction execution by exploiting instruction parallelism with

		the reasons one of ordinary skill in the art would have been motivated to select the references and combine them.	superscalar processors to increase performance (Hilgendorf, column 1, line 56-column 2, line 63). Martin has specifically taught a circuit that is configured to detect optimizable sequences of instruction codes on-the-fly (Martin abstract column 2 lines 20-52). It would have been obvious to one of ordinary skill in the art at the time the invention was made to have the circuit of Hilgendorf be configured to detect optimizable sequences of instruction codes on-the-fly. Since Martin shows us that one instruction takes the place of two or more instructions, and that this function improves performance (Martin column 2 lines 20-30), by allowing other locations in the instruction buffer to be open, and by allowing for a single instruction to take the place of multiple instructions, the processor would also save time (Martin column 2 lines 20-30). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have the processor of Hilgendorf achieve its goal of improving system performance by configuring the circuit to detect optimizable sequences of instruction codes on-the-fly, as taught by Martin (Martin column 2 lines 20-30). Therefore this argument is moot.
Rejection of Claim 15	Pages 22-24	Specifically, the Examiner failed to present evidence or provide a convincing line of reasoning explaining exactly how Hilgendorf's invention could be used with Java code. In particular, Hilgendorf's invention is directed to dynamic conversion between different instruction codes by recombination of instruction elements. However, Hilgendorf appears to be silent	*"The test for obviousness is not whether the features of a secondary reference may be bodily incorporated into the structure of the primary reference.... Rather, the test is what the combined teachings of those references would have suggested to those of ordinary skill in the art." In re Keller, 642 F.2d 413, 425, 208 USPQ 871, 881 (CCPA 1981). See also In re Sneed,*

	regarding converting and executing Java code. ... Hilgendorf is directed to rearranging elements of one instruction code to generate another instruction code. The Examiner has failed to present any evidence or convincing line of reasoning to factually support a conclusion that Jave bytecodes can merely be arranged, as performed by Hilgendorf, to generate host instructions. ... Because the Examiner has failed to make the required showing that there is a reasonable expectation of success in using the invention of Hilgendorf with Java bytecodes, the Examiner failed to factually establish a prima facie conclusion of obviousness.	*710 F.2d 1544, 1550, 218 USPQ 385, 389 (Fed. Cir. 1983) ("[I]t is not necessary that the inventions of the references be physically combinable to render obvious the invention under review."); and In re Nievelt, 482 F.2d 965, 179 USPQ 224, 226 (CCPA 1973) ("Combining the teachings of references does not involve an ability to combine their specific structures."). "MPEP 2145 III.* In this case, instruction translation is taught by Hilgendorf, which is the primary reference. The secondary reference is information available to one of ordinary skill in the art, which is supported by extrinsic evidence cited in Gee, Wilmot, II, Favor, and Aravamudan et al. Hilgendorf has taught the apparatus of claim 1. Hilgendorf has not explicitly taught wherein said instruction codes of said first instruction set comprise Java bytecodes. However, Hilgendorf has taught that the translation circuit translates the first instruction code into many different codes by replacing the translation table's contents (Hilgendorf column 3 lines 45-56, column 4 lines 51-61). One such possible instruction code is Java. The Java language has been quickly embraced by the computer software community and has been demonstrated to be efficient and robust for a wide variety of general purpose computing applications (Gee, Column 2, lines 28-37). For example, Hilgendorf's invention is part of a superscalar processor. Superscalar processors are

		typically employed as webs servers because of the high performance associated with the superscalar design (For extrinsic evidence of this fact, please see US Patent 6,615,340, Wilmot, II, column 1, lines 10-23, and US Patent 5,926,642, Favor, Columns 60-62, specifically see column 62, lines 41-45.). A common way of executing programs on the world wide web is by using java bytecodes because using java bytecodes enables the execution of programs across any platform so long as an interpreter, or translator, is present (For extrinsic evidence of this fact, please see US Patent 6,502,109, Aravamudan et al., column 2, lines 45-65). Since superscalar computers are typically employed as web servers, one of ordinary skill in the art at the time the invention was made would have recognized to employ Hilgendorf's superscalar computer as a web server because of the high performance associated with the superscalar design. Furthermore since Java is a commonly used programming language for the execution of programs on the world wide web, one of ordinary skill in the art at the time the invention was made would have recognized the desirability of executing Java bytecodes on a web server that includes Hilgendorf's invention since most of the programs available on the world wide web are already written in Java bytecodes. In order to execute these widely available Java bytecodes in Hilgendorf's superscalar processor, the Java bytecodes must be translated to the host's instruction code. Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to have the

			superscalar web server of Hilgendorf translate Java bytecodes into the host's computer code since Java is a popular and widely used programming language, and many programs available on the world wide web would already be written in Java bytecodes. It is not necessary for the Examiner to explain the exact design of the combined teachings to yield a proper obvious combination of teachings. Rather, the test is what the combined teachings of those references would have suggested to those of ordinary skill in the art. In this case, the combined teachings would have suggested to one of ordinary skill in the art at the time this invention was made to have the claimed first instruction set translated using the instruction translator of Hilgendorf specifically be Java bytecodes, see the rejection above. Therefore this argument is moot.
Rejection of Claim 16	Pages 24-26	Nowhere in the cited text does Gee expressly mention a circuit comprises a portion of a Java virtual machine implemented in hardware, as presently claimed.	However, Gee has taught that implementing a Java Virtual Machine in hardware (i.e. a direct execution JAVA processor) reduces the drawbacks of a JVM not implemented in hardware, such as speed and difficulty in meeting strict timing requirements. (Gee, column 2, lines 38-47) Thus allowing for a higher instruction throughput, and the ability to complete instructions more quickly, which is important with real-time applications. So Gee has in fact taught a portion of a Java virtual machine implemented in hardware, as presently claimed. Therefore this argument is moot.
Rejection of Claim	Pages 26-27	Nowhere in the above text does Gee expressly mention a sequence of instruction codes of an	However, Gee has taught a Java Virtual Machine, or JVM. A JVM by definition uses stack based

21		instruction set comprises one or more virtual stack references, as presently claimed.	processing (Gee, column 7, lines 1-7, column 8, lines 47-49). Since the JVM uses stack based processing, the JVM must necessarily have the Java instruction codes comprise one or more virtual stack references in order to properly execute and/or process instructions using the stack. So Gee has in fact taught a sequence of instruction codes of an instruction set comprises one or more virtual stack references, as presently claimed. Therefore this argument is moot.
Rejection of Claim 21	Page 27	The Examiner failed to present any evidence or convincing line of reasoning regarding a reasonable expectation of success.	It is not necessary for the Examiner to explain the exact design of the combined teachings to yield a proper obvious combination of teachings. Rather, the test is what the combined teachings of those references would have suggested to those of ordinary skill in the art. In this case, the combined teachings would have suggested to one of ordinary skill in the art at the time the invention was made to have the superscalar web server of Hilgendorf translate Java bytecodes into the host's computer code using a flexible, run-time, Java Virtual Machine since Java is a popular and widely used programming language, and many programs available on the world wide web would already be written in Java bytecodes (also see the rejection of claim 21 above). When the Hilgendorf reference translates Java code with a Java Virtual Machine, as taught by Gee, the sequences of instruction codes of said instruction set necessarily comprise one or more virtual stack references (Gee, Column 7, lines 1-10) because in order to implement Java with a Java Virtual Machine

		it is necessary to have virtual stack references. Applicant has not provided any arguments or evidence to explain why there would not be a reasonable expectation of success. Examiner finds no reasoning why there would not be a reasonable expectation of success to combine Hilgendorf and Gee to arrive at the claimed invention in claim 21. It is entirely reasonable to have the sequences of instruction codes of said instruction set comprise one or more virtual stack references, as explained above. Therefore this argument is moot.

(11) Related Proceeding(s) Appendix

No decision rendered by a court or the Board is identified by the examiner in the Related Appeals and Interferences section of this examiner's answer.

For the above reasons, it is believed that the rejections should be sustained.

Respectfully submitted,

Tonia L. M. Dollinger

February 7, 2008

Conferees:

ALFORD KINDRED
SUPERVISORY PATENT EXAMINER

NABHAN
MANO PADMANABHAN
TC 2100